UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/717,941	11/20/2003	Stephen La Roux Blinick	TUC920030135US1	9013

45216 7590 12/20/2006

KUNZLER & ASSOCIATES
8 EAST BROADWAY
SUITE 600
SALT LAKE CITY, UT 84111

EXAMINER	
WANG, BEN C	
ART UNIT	**PAPER NUMBER**
2196	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	12/20/2006	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

B1

Office Action Summary	Application No.	Applicant(s)	
	10/717,941	BLINICK ET AL.	
	Examiner	Art Unit	
	Ben C. Wang	2196	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE ____ MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *20 November 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☒ Claim(s) *1* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-30 are pending in this application and presented for examination.

Specification *Objections*

2. The specification is objected to because the following informalities:

"persistent data the should be make available to new code image", cited in [0014], line 13, should be corrected as "persistent data that should be make available to new code image".

Appropriate correction is required.

Claim Objections

3. Claims 1 is objected to because the following informalities:

"a copy module configure to copy", claim 1, line 8, should be corrected as "a copy module configured to copy".

Appropriate correction is required.

Claim Rejections – 35 USC § 103(a)

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. Claims 1-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Talati (hereafter 'Talati') (Pub. No. US 2004/0044997 A1) in view of Hiller et al. (hereafter 'Hiller') (Patent No. US 6,658,659 B2).

6. **As to claim 1**, Talati discloses an apparatus for updating a code image (Fig. 2), comprising of a loader configured to load a new code image ([0014], lines 2-3; Fig. 2, element 102; [0047], lines 1-2) into a temporary memory location (Fig. 1, element 108; [0009], staging area) separate from a memory space (Fig. 1, element 110; [0009], line 2, runtime area) occupied by and used by an old code image (Fig. 2, elements 218, 110; [0046], lines 1-8; [0047], lines 1-2) and a copy module configure to copy the new code image into the memory space occupied by the old code image (Fig. 2, element 202; Fig. 3, element 302; [0013], lines 1-3).

But, Talati does not disclose a conversion module configured to selectively reconcile incompatibilities between the old code image and the new code image.

However, in an analogous art, Hiller discloses a conversion module configured to selectively reconcile incompatibilities between the old code image and the new code image (Fig. 2A, element 206; Fig. 2B, elements 102, 218; Col. 3, lines 42-44; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance compatible version module loading in Talati system.

The motivation is that a version aware conversion module will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

1. **As to claim 10**, Talati discloses an apparatus for updating a code image (Fig. 2, copier copies new code), comprising of an update module configured to load a new code image ([0014], lines 2-3; Fig. 2, element 102; [0047], lines 1-2) into a temporary memory location (Fig. 1, element 108; [0009], line 2, staging area) separate from a memory space occupied by and used by an old code image (Fig. 2, elements 218, 110; [0046], lines 1-8) and a bootstrap module within the new code image that executes subsequent to the update module (Fig. 2, element 202; [0047], lines 1-2).

But, Talati does not disclose the bootstrap module configured to selectively reconcile incompatibilities between the old code image and the new code image prior to copying the new code image into the memory space occupied by the old code image.

However, in an analogous art, Hiller discloses the bootstrap module configured to selectively reconcile incompatibilities between the old code image and the new code image prior to copying the new code image into the memory space occupied by the old code image (Fig. 2A, element 206; Fig. 2B, elements 252, 258 and 260; Col. 3, lines 42-44; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of

Hiller to enhance the bootstrap module configured with compatible version module loading capability in Talati system.

The motivation is that a version aware bootstrap module will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

As to claim 13, Talati discloses a system that overlays an old code image with a new code image with minimal interruption of operations being performed by execution of the old code image ([0011]), the system comprising of a memory comprising an old code image (Fig. 1, element 110; Fig. 2, element 108) and a buffer (Fig. 1, element 108; Fig. 2, element 110) configured to store a new code image; a processor executing instructions of the old code image to perform one or more operations (Fig. 1, element 106), the processor configured to execute instructions of the old code image and the new code image ([0011]; [0032], lines 1-5); a data structure configured to store an old code image pointer (Fig. 2, elements 204, 208, and 212; [0023], lines 4-10) and a new code image pointer (Fig. 2, elements 206, 210, and 214; [0023], lines 11-16); wherein, in response to an interrupt, the processor begins executing bootstrap code within the new code image ([0040]).

But, Talati does not disclose the bootstrap code configured to reconcile incompatibilities between the old code image and the new code image.

However, in an analogous art, Hiller discloses the bootstrap code configured to reconcile incompatibilities between the old code image and the new code image (Fig.

2A, element 206; Fig. 2B, elements 252, 258 and 260; Col. 3, lines 42-44; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the bootstrap code configured to reconcile incompatibilities between the old code image and the new code image in Talati system.

The motivation is that a version aware bootstrap code configured to reconcile incompatibilities will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

2. **As to claim 20**, Talati discloses a method for updating a code image (Fig. 2, Copier copies new code), comprising of loading a new code image into a temporary memory location (Fig. 1, element 108; [0009], staging area) separate from a memory space (Fig. 1, element 110; [0009], line 2, runtime area) occupied by and used by an old code image (Fig. 2, elements 218, 110; [0046], lines 1-8; [0047], lines 1-2); copying the new code image into the memory space occupied by the old code image (Fig. 2, element 202; Fig. 3, element 302).

But, Talati does not disclose a conversion module configured to selectively reconcile incompatibilities between the old code image and the new code image.

However, in an analogous art, Hiller discloses selectively reconciling incompatibilities between the old code image and the new code image (Fig. 2A, element 206; Fig. 2B, elements 252, 258 and 260; Col. 3, lines 42-44; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the functionality of selectively reconciling incompatibilities between the old code image and the new code image in Talati system.

The motivation is that a selectively reconciling incompatibilities will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

3. **As to claim 29**, Talati discloses an apparatus for updating a code image ([0014], lines 2-3; Fig. 2, copier copies new code; [0047], lines 1-2), the apparatus comprising of means for loading a new code image (Fig. 2, element 102) into a temporary memory location (Fig. 1, element 108; [0009], staging area) separate from a memory space (Fig. 1, element 110) occupied by and used by an old code image; means for copying the new code image into the memory space occupied by the old code image (Fig. 2, element 202; Fig. 3, element 302).

But, Talati does not disclose a conversion module configured to selectively reconcile incompatibilities between the old code image and the new code image.

However, in an analogous art, Hiller discloses means for selectively reconciling incompatibilities between the old code image and the new code image (Fig. 2A, element 206; Fig. 2B, elements 252, 258 and 260; Col. 3, lines 42-44; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the means for compatible version module loading in Talati system.

The motivation is that means for selectively reconciling incompatibilities will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

4. **As to claim 30**, Talati discloses an article of manufacture comprising a program storage medium readable by a processor and embodying one or more instructions executable by a processor to perform a method for updating a code image (Fig. 1), the method comprising of loading a new code image into a temporary memory location separate from a memory space occupied by and used by an old code image (Fig. 2, elements 218, 110; [0046], lines 1-8; [0047], lines 1-2); copying the new code image into the memory space occupied by the old code image (Fig. 2, element 202; Fig. 3, element 302).

But, Talati does not disclose a conversion module configured to selectively reconcile incompatibilities between the old code image and the new code image.

However, in an analogous art, Hiller discloses the method of selectively reconciling incompatibilities between the old code image and the new code image (Fig. 2A, element 206; Fig. 2B, elements 102, 218; Col. 3, lines 42-44; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of

Hiller to enhance the method of selectively reconciling incompatibilities between the old code image and the new code image in Talati system.

The motivation is that the method of selectively reconciling incompatibilities will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

5. **As to claim 2**, Talati discloses the apparatus wherein the old code image is updated substantially concurrent with normal execution of transactions by the apparatus ([0006]; [0008], lines 5-12]; [0011]).

6. **As to claim 3**, Talati discloses the apparatus further comprising an initialization module configured to initiate execution of a run-time segment of the new code image ([0014]).

7. **As to claim 4**, Talati discloses the apparatus wherein the copy module copies the new code image into the memory space in response to a load request (Fig. 2, Copier copies new code; Fig. 3, element 302).

But, Talati does not disclose the apparatus wherein in response to reconciliation of the incompatibilities.

However, in an analogous art, Hiller discloses the apparatus wherein in response to reconciliation of the incompatibilities (Col. 3, lines 42-44; Col. 4, lines 57-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the apparatus wherein in response to reconciliation of the incompatibilities in Talati system.

The motivation is that the apparatus of wherein in response to reconciliation of the incompatibilities will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

8. **As to claim 5**, Talati does not disclose the apparatus further comprising a logic module configured to access version information for the old code image and version information for the new code image and identify an incompatibility based at least in part on a difference between the version information.

However, in an analogous art, Hiller discloses the apparatus further comprising a logic module configured to access version information for the old code image and version information for the new code image and identify an incompatibility based at least in part on a difference between the version information (Fig. 2A, elements 202, 204, 208, 210, 212, and 206; Fig. 2B; Col. 7, lines 31-35).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the apparatus comprising such a logic module to identify incompatibility functionality in Talati system.

The motivation is that the apparatus comprising such a logic module to identify incompatibility functionality will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

9. **As to claim 6**, Talati does not disclose the apparatus wherein the conversion module is configured to update modules that interface with the new code image based at least in part on a difference between the version information.

However, in an analogous art, Hiller discloses the apparatus wherein the conversion module is configured to update modules that interface with the new code image based at least in part on a difference between the version information (Fig. 3A; Fig. 3B; Fig. 3C; Col. 8, lines 18-23, lines 31-34, lines 49-56; Fig. 4; Col. 10, lines 24-26).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the apparatus wherein the conversion module is configured to update modules that interface with the new code image based at least in part on a difference between the version information in Talati system.

The motivation is that the apparatus of a version aware conversion module will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

10. **As to claim 7**, Talati discloses recognizing persistent data associated with the old code image and associates the persistent data with the new code image such that the persistent data is available in response to execution of the run-time segment of the new code image ([00110; [0012] – the conversion module provides two separate functionalities (a) to selectively reconcile incompatibilities between the old code image and the new code image cited in claim 1; and, (b) to recognize persistent data described in this claim).

11. **As to claim 8**, Talati does not disclose the apparatus wherein at least one of the incompatibilities comprises different initialization requirements.

However, in an analogous art, Hiller discloses the apparatus wherein at least one of the incompatibilities comprises different initialization requirements (Fig. 3B; Fig. 3C; Fig. 5, element 502; Col. 11, line 62 through Col. 12, line 4; Col. 12, lines 25-32; Col. 9, lines 22-28).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to possibly have incompatible modules in Talati system.

The motivation is to have an apparatus with version aware loader which will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

12. **As to claim 9**, Talati does not disclose the apparatus wherein at least one of the incompatibilities comprises a difference between data structures used by the old code image and data structures used by the new code image.

However, in an analogous art, Hiller discloses the apparatus wherein at least one of the incompatibilities comprises a difference between data structures used by the old code image and data structures used by the new code image (Fig. 3B; Fig. 3C; Fig. 5, element 502; Col. 11, line 62 through Col. 12, line 4; Col. 12, lines 25-32; Col. 9, lines 22-28).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to possibly have incompatible modules in Talati system.

The motivation is to have an apparatus wherein a version aware loader which will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

13. **As to claim 11**, Talati does not disclose the apparatus wherein the bootstrap module comprises a conversion module configured to reconcile the incompatibilities base on version information for the old code image and the new code image and a copy module configured to copy the new code image over the old code image in response to reconciliation of the incompatibilities.

However, in an analogous art, Hiller discloses the apparatus wherein the bootstrap module comprises a conversion module configured to reconcile the

incompatibilities base on version information for the old code image and the new code image (Fig. 2B, elements 252, 258, and 260; Fig. 4, step 410; Col. 10, lines 31-38) and a copy module configured to copy the new code image over the old code image in response to reconciliation of the incompatibilities (Fig. 2B, step 260; Fig. 4, step 412; Col. 10, lines 24-26).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the apparatus wherein the bootstrap module comprises a conversion module configured to reconcile the incompatibilities base on version information for the old code image and the new code image and a copy module configured to copy the new code image over the old code image in response to reconciliation of the incompatibilities in Talati system.

The motivation is to have an apparatus wherein a version aware bootstrap and copy modules that will ensure that loaded software modules are compatible with one another before copying them and therefore will execute properly ([Hiller], Abstract, lines 10-13).

14. **As to claim 12**, Talati does not disclose the apparatus wherein at least one of the incompatibilities comprises a difference between data structures used by the old code image and data structures used by the new code image.

However, in an analogous art, Hiller discloses the apparatus wherein at least one of the incompatibilities comprises a difference between data structures used by the old

code image and data structures used by the new code image (Fig. 3B; Fig. 3C; Fig. 5, element 502; Col. 11, line 62 through Col. 12, line 4; Col. 12, lines 25-32; Col. 9, lines 22-28).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to possibly have the apparatus wherein incompatible data structures within modules in Talati system.

The motivation is to have the apparatus wherein incompatible data structures within modules will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

15. **As to claim 14**,Talati does not disclose the system wherein the bootstrap code overlays the new code image in memory with the old code image in response to reconciliation of the incompatibilities.

However, in an analogous art, Hiller discloses the system wherein the bootstrap code overlays the new code image in memory with the old code image in response to reconciliation of the incompatibilities (Fig. 2A, element 206; Fig. 2B; Col. 3, lines 42-46; Col. 4, lines 64-67).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the system wherein the bootstrap code overlays the new code image

in memory with the old code image in response to reconciliation of the incompatibilities in Talati system.

The motivation is that the system wherein the version aware bootstrap code will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

16. **As to claim 15**, Talati discloses the system wherein in response to the interrupt, the processor executes an update module of the old code image that loads the new code image into the buffer (Fig. 3, step 302; [0040]; [0041]).

17. **As to claim 16**, Talati discloses the system wherein the update module stores the old code image pointer (Fig. 2, elements 204, 208, and 212; [0023], lines 4-10) and the new code image pointer (Fig. 2, elements 206, 210, and 214; [0023], lines 11-16) in the data structure.

18. **As to claim 17**, Talati discloses the system of wherein the update module reads a new code image header identified by the new code image pointer (Fig. 2, elements 206, 210; [0034]) to determine the location of the bootstrap code within the new code image ([0037], lines 5-7).

19. **As to claim 18**, Talati does not disclose the system of wherein the bootstrap code identifies incompatibilities by reviewing capability fields of the old code image.

However, in an analogous art, Hiller discloses the system of wherein the bootstrap code identifies incompatibilities by reviewing capability fields of the old code image (Fig. 5, element 502; Col. 4, lines 45-51; Col. 9, lines 22-28; Col. 12, lines 25-39)).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the system of wherein the bootstrap code identifies incompatibilities by reviewing capability fields of the old code image in Talati system.

The motivation is that the system with a version aware bootstrap code will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

20. **As to claim 19**, Talati does not disclose the system wherein the bootstrap code identifies incompatibilities by comparing version information of the new code image with version information of the old code image.

However, in an analogous art, Hiller discloses the system wherein the bootstrap code identifies incompatibilities by comparing version information of the new code image with version information of the old code image.

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the system wherein the bootstrap code identifies incompatibilities by

comparing version information of the new code image with version information of the old code image in Talati system.

The motivation is that the system wherein a version aware bootstrap code will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

21. **As to claim 21**, Talati discloses the method wherein the old code image is updated substantially concurrently with execution of regular computer operations ([0011]; [0014]).

22. **As to claim 22**, Talati disclose the method further comprising initiating execution of a run-time segment of the new code image ([0049]).

23. **As to claim 23**, Talati does not disclose the method wherein the new code image is not copied into the memory space until incompatibilities are reconciled.

However, in an analogous art, Hiller discloses the method wherein the new code image is not copied into the memory space until incompatibilities are reconciled.

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the method wherein the new code image is not copied into the memory space until incompatibilities are reconciled in Talati system.

The motivation is that the method wherein the version aware bootstrap code will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

24. **As to claim 24**, Talati does not disclose the method further comprises accessing capability information for the old code image and capability information for the new code image and identifying an incompatibility based at least in part on a difference between the capability information.

However, in an analogous art, Hiller discloses the method further comprises accessing capability information for the old code image and capability information for the new code image and identifying an incompatibility based at least in part on a difference between the capability information (Fig. 5, element 502; Col. 12, lines 25-32; Col. 13, lines 12-19).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to enhance the method further comprises accessing capability information for the old code image and capability information for the new code image and identifying an incompatibility based at least in part on a difference between the capability information in Talati system.

The motivation is that the method wherein the version aware bootstrap code will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

25. **As to claim 25**, Talati does not disclose the method further comprising updating modules that interface with the new code image based at least in part on a difference between the capability information.

However, in an analogous art, Hiller discloses the method further comprising updating modules that interface with the new code image based at least in part on a difference between the capability information (Fig. 5, element 502; Col. 12, lines 25-32; Col. 13, lines 12-19).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to possibly have different capability information with the new code image in Talati system.

The motivation is to have the method of a version aware loader which will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

26. **As to claim 26**, Talati discloses the method further comprising determining persistent data associated with the old code image and associating the persistent data with the new code image such that the persistent data is available in response to execution of a run-time segment of the new code image ([00110; [0012]).

27. **As to claim 27**, Talati does not disclose the method wherein at least one of the incompatibilities comprises different initialization requirements.

However, in an analogous art, Hiller discloses the method wherein at least one of the incompatibilities comprises different initialization requirements (Fig. 3B; Fig. 3C; Fig. 5, element 502; Col. 11, line 62 through Col. 12, line 4; Col. 12, lines 25-32; Col. 9, lines 22-28).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to possibly have incompatible methods in Talati system.

The motivation is that the method of a version aware loader that will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

28. **As to claim 28**, Talati does not disclose the method wherein at least one of the incompatibilities comprises a difference between data structures used by the old code image and data structures used by the new code image.

However, in an analogous art, Hiller discloses the method wherein at least one of the incompatibilities comprises a difference between data structures used by the old code image and data structures used by the new code image (Fig. 3B; Fig. 3C; Fig. 5, element 502; Col. 11, line 62 through Col. 12, line 4; Col. 12, lines 25-32; Col. 9, lines 22-28).

Therefore, it would have been obvious to one of ordinary skill in the art, at the time the invention was made to combine the teachings of Talati and the teachings of Hiller to possibly have the method wherein incompatible data in Talati system.

The motivation is to have a version aware loader which will check and ensure that loaded software modules are compatible with one another and will therefore execute properly ([Hiller], Abstract, lines 10-13).

Conclusion

29. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Nabil El-Hady can be reached on 571-272-2333. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO

Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

BCW

November 27, 2006

NABIL M. EL-HADY
SUPERVISORY PATENT EXAMINER